Vol. 48 No. 6 IPSJ Journal June 2007

*Regular Paper*

# Anomaly Detection Using Integration Model of Vector Space and Network Representation

Mizuki Oka[†,☆] and Kazuhiko Kato[†]

We propose the Eigen Co-occurrence Matrix (ECM) method, which is a modeling method for tracking the behaviors of an individual, system, or network in terms of event sequences of discrete data. Our method uses the correlation between events in a sequence to extract distinct characteristics. A key idea behind the ECM method is to regard a sequence as a serialized sequence that originally had structural relations and to extract the embedded dependencies of the events. To test its retrieval performance, we applied the ECM method to the problem of anomaly detection in intrusion detection systems. Specifically, we used the method to model a UNIX command sequence and attempted to detect intruders masquerading as valid users. The experimental results reveal that the ECM method offers distinct characteristic models for analyzing event sequences.

## 1. Introduction

The problem of anomaly detection in intrusion detection systems can be examined by modeling the behaviors of an individual, system, or network in terms of event sequences of discrete data[1]. The created models are then used to classify normal and abnormal behaviors by computing similarities. Approaches to anomaly detection differ in how they create *models* and how they define *similarity*. Various modeling approaches for event sequences have been proposed in intrusion detection systems literature. Typical instances of such approaches can be classified as either *vector space*-based methods[2]~[4] and the *network*-based methods[5]~[8].

The advantages of these two types of methods are complementary to each other. The vector space-based methods can automatically generate a model from an event sequence, but the relations between the events cannot be represented, whereas the network-based methods can represent the relations between the events, but a domain specific knowledge is often required to define the topology of the network.

This paper describes a method that has the advantages of both of the above-mentioned methods. The idea behind the method is to regard an event sequence as a serialized sequence that originally had structural relations and to extract the embedded dependencies of the events. Logging operations in most computer systems log a series of actions of a task in a serialized manner. Consider a situation wherein a UNIX user uses a shell to perform a task $A$ using commands logged as $(a_1, a_2, a_3, \ldots, a_n)$. Assume that there is a structural dependency from $a_1$ to $a_3$, meaning that the user had to perform action $a_1$ to perform action $a_3$. However, the serialized command log would not explicitly describe this dependency. Our method automatically creates a model that extracts these embedded structural relations among events.

Our method, which is called the Eigen Co-occurrence Matrix (ECM) follows the previously reported terminology[9] and mainly uses two techniques, namely a co-occurrence matrix and a principal component analysis (PCA). The concept of the co-occurrence matrix, which often appears in the field of information retrieval, is to represent relations between words extracted from one or several documents. These matrices are often used as a preprocessing for feature extraction from text documents. The ECM method, on top of a conventional matrix, uses sequential information to construct a co-occurrence matrix so that embedded structural relations among events in a sequence are extracted.

Co-occurrence matrices, depending on the number of observed events, can be sparse and highly dimensional. To reduce the dimensionality and to extract its principal features, the ECM method uses PCA. PCA is a common statistical technique used for finding patterns in high dimensional data and successfully used in applications like facial recognition and image

† University of Tsukuba
☆ Presently with Japan Society for the Promotion of Science

compression.

Throughout the development of the ECM method, attention was paid to the seamless integration of the vector space- and network-based methods. Consequently, the method can easily extract statistically sound structural relations, and the resulting model can be used to analyze the characteristic features of a sequence. Moreover, the models can be used to measure similarities among sequences for classifying anomalous sequences from normal sequences.

The remainder of this paper is organized as follows. Section 2 describes related work and Section 3 describes the basic concepts and terminology used in this paper. Section 4 proposes our method for modeling sequences. Section 5 describes anomaly detection using ECM models. Section 6 reports our experiments with the masquerader dataset using UNIX command sequences and shows the results. We conclude this paper in Section 7.

## 2. Related Work

Most of the methods used in analyzing sequences for anomaly detection can be categorized into two groups, *vector space*-based and *network*-based.

One example of a *vector space*-based method is n-gram, a technique often used in the field of classification. This technique models sequences by counting the occurrences of n-connected events (n-gram). However, unlike the ECM method, this method does not consider the correlations between events that are not adjacent to each other, in which there may be implicit relations.

One example of the *network-based* method is Sekar, et al.'s[5] finite state automaton (FSA), a profile of a program from program-generated system calls. A more accurate profile of a program is generated by using the pushdown automaton developed by Wagner, et al.[6]. The use of such automaton-based models is adequate when analyzing sequences that have well-defined syntax, such as programs, because the nodes in the automaton can remember the short and long-range relations of events by constructing the language of the sequences.

As an alternative approach for a network-based model, Warrender, et al.[10] used the hidden Markov model (HMM). An HMM models sequences based on a predefined topology, on which the probability of occurrences and transition probability among the observed events are learned. The learned HMM computes the probability of an event in an input sequence. While the methods described above learn and compute event probabilities on top of a predefined network, the ECM method automatically generates networks with an arbitrarily chosen topology in which the characteristic binomial relations of events of an input sequence are represented. Moreover, the generated networks can be used to classify the input sequence.

## 3. Basic Concepts and Terminologies

Before describing our proposed method, we explain the basic concepts and terminologies used in the method, namely, *window*, *co-occurrence matrix*, and *network model*.

### 3.1 Window

Let $X = (x_1, x_2, \ldots, x_m)$ be a sequence of events in a set $E = \{e_1, e_2, \ldots, e_n\}$, where each event has an associate time of occurrence denoted by $T = (t_1, t_2, \ldots, t_m)$. **Figure 1** describes an example sequence.

A sequence is analyzed per *window*, a unit of analysis that slides over specified segments on the sequence. In Fig. 1, a window with a size of five is depicted, thus making two windows. The size of a window which we denote as $w$ is an important factor in analyzing a sequence. A window may or may not have overlaps, and the size can usually be determined through trial and error and is empirically based on detection performance.

### 3.2 Co-occurrence Matrix

Most logging operations in computer systems log a series of actions as a serialized sequence. A *co-occurrence matrix* is computed from such a serialized sequence, intending to retrieve embedded correlations among events. Here, an event indicates a logging unit in a sequence.

Given sequence $X$, a correlation is considered for every event pair, $e_i$ and $e_j$ in event set $E$. We would like the correlation of the two events to be stronger when the interval time between

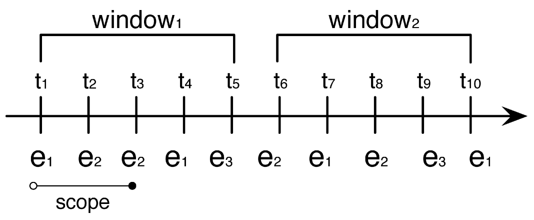


**Fig. 1** Window in sequence.

```
$C_{i,j}$ : correlation strength between $e_i$ and $e_j$
$s$ : scope size
for $i = 1$ to $m$ do
  for $j = 1$ to $m + s$ do
    $C_{(i,j)} = 0$
  end for
end for
for $p = 1$ to $n$ do
  for $q = p$ to $(p + s)$ do
    $i$ = index of $e_p$
    $j$ = index of $e_q$
    $C_{(i,j)} = C_{(i,j)} + 1$
  end for
end for
```

**Fig. 2** Pseudo-code for constructing a co-occurrence matrix from a sequence.

the two events is short and/or when they appear more frequently. We compute such correlation strength $C_{(i,j)}$ between $e_i$ and $e_j$ by counting their number of occurrences within a certain interval which we call *scope*. In Fig. 1, a scope with a size of two is described. As with window size, the size of the scope needs to be determined by trial and error, and is empirically based on retrieval performance.

Computing the correlation strength for every event pair in $E$ in the sequence $X$ generates a *co-occurrence matrix*, which we denote as $C_x$. The algorithm for constructing the co-occurrence matrix is described in **Fig. 2**. The resulting co-occurrence matrix for the first window of the sequence in Fig. 1 is shown in Fig. 3.

### 3.3 Network Model

The co-occurrence matrix described above can be represented as a directed graphs by regarding it as an adjacency matrix. We call such directed graph *network models*. An example network model derived from the co-occurrence matrix shown in **Fig. 3** is depicted in **Fig. 4**. A network model is essentially a directed graph of the co-occurrence matrix (or adjacency matrix in other words) with rows and columns labeled by a set of events in $E$. For example, the value 2 in the second row of the first column in the matrix of Fig. 3 represents the correlation strength from event $e_2$ to $e_1$ in the Fig. 4.

An adjacency matrix, $C$, can contain both positive and negative values, which enables us to divide $C$ into $C_+$ and $C_-$ as follows,

$$C = C_+ + C_-, \qquad (1)$$

where $C_+$ (or $C_-$) denotes an adjacency matrix whose elements are determined by the corresponding positive (or negative) elements in $C$.

$$\begin{array}{c|ccc} & e_1 & e_2 & e_3 \\ \hline e_1 & 0 & 2 & 1 \\ e_2 & 2 & 1 & 1 \\ e_3 & 0 & 0 & 0 \end{array}.$$

**Fig. 3** Co-occurrence matrix for sequence in Fig.1.

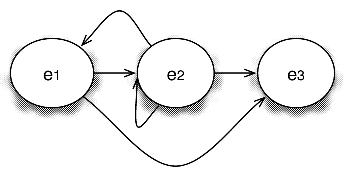


**Fig. 4** Network model of a co-occurrence matrix.

A network model can be constructed from $C_+$ (or $C_-$) correspondingly. We call the resulting model a *positive network* (or *negative network*).

## 4. Eigen Co-occurrence Matrix Method

This section presents the ECM method that uses co-occurrence matrices and PCA to model a sequence. A notable feature of the method is its automatic generation of a model that represents implicit structural relations of events; the structural relations are represented in the form of *network models*, which gives a good idea of the overall behavior of an input sequence. The overall procedure of the method is shown in **Fig. 5**.

### 4.1 Principal Component Analysis

Principal componenet analysis (PCA) is a process whereby a data set expressed in M-dimensional space is reduced to a K dimensional space. In this space, the K dimensions computed represent the K-axis subspace of the original data set, which accounts for as much of the variation in the data set as possible and thus is optimal for moving about the given data set in K dimensions.

PCA is necessary to our method because a co-occurrence matrix of a sequence with $n$ unique events has $n \times n$ degrees of freedom. Without PCA, for example, 500 unique events would require calculations in 250,000-dimensional space. This is computationally unreasonable. By using PCA, this 250,000-dimensional space can be reduced to an arbitrarily small coordinate space, with each of the resulting axes expressing the most important aspects of co-occurrences.

Let a set of sequences be $X = \{X_1, X_2, \ldots, X_N\}$ and their corresponding co-occurrence matrices be $C = \{C_1, C_2, \ldots, C_N\}$. We sub-

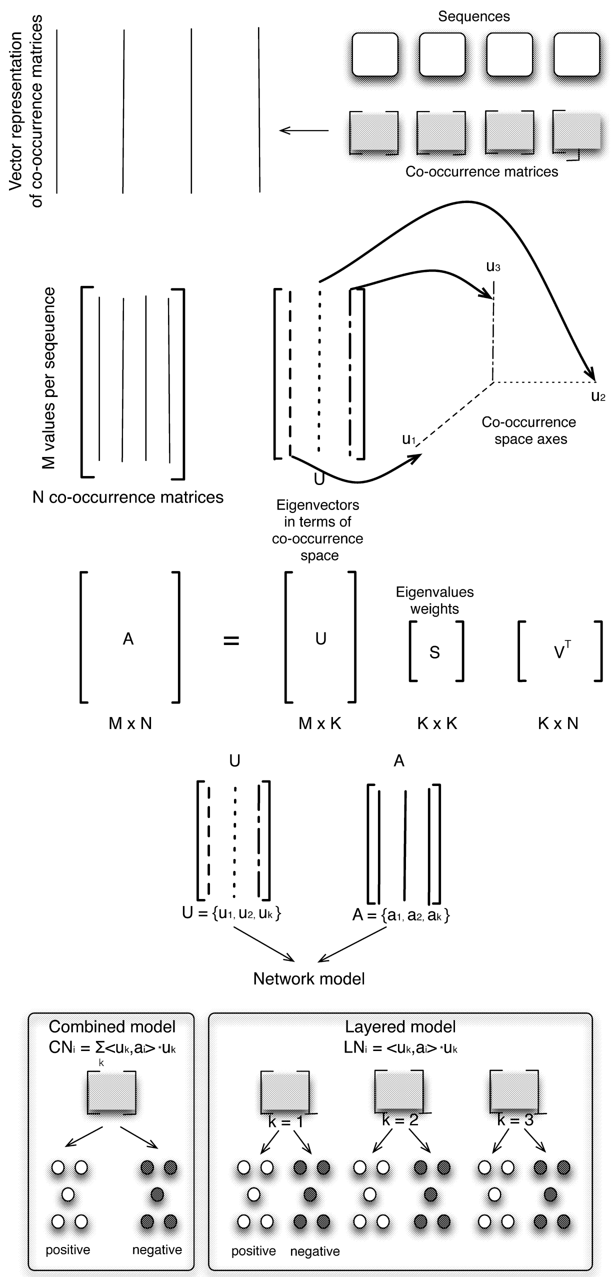


**Fig. 5** Overview of Eigen Co-occurrence Matrix Method.

tract their average co-occurrence matrix $C_{mean}$ from each matrix and represent them in vectors by concatenating each row of matrices and denote them as $A = \{a_1, a_2, \ldots, a_N\}$. PCA is done by performing singular value decomposition (SVD) upon the dataset $\{a_1, a_2, \ldots, a_N\}$. This generates a set of three matrices, U, S, and V. $U$ is an $M \times K$ matrix, where $M$ is $n \times n$ degree. The columns of $U$ are a list of normalized eigenvectors of covariance matrix $AA^T$ representing the axes of greatest variance in terms of $M$ dimensions of vector representation of co-occurrence matrices after subtraction of mean matrix $C_{mean}$. These eigenvectors represent the new coordinate space. $S$ is a $K \times K$ diagonal matrix whose values on the diagonals represent the square root values for each of the eigenvalues in matrix $AA^T$. A larger weight equals a greater importance of eigenvalues thus it allows us to rank the eigenvalues in order of impor-

$$\begin{array}{l}
 & \\
\end{array}
\begin{array}{lccccccc}
 & \textit{emacs} & \textit{latex} & \textit{bibtex} & \textit{xdvi} & \textit{cd} & \textit{ls} & \textit{less} \\
\textit{emacs} & 1 & 2 & 2 & 0 & 0 & 0 & 0 \\
\textit{latex} & 0 & 1 & 1 & 2 & 0 & 0 & 0 \\
\textit{bibtex} & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
\textit{xdvi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{cd} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{ls} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{less} & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}$$

**Fig. 6** Co-occurrence matrix for $X_1$.

$$\begin{array}{lccccccc}
 & \textit{emacs} & \textit{latex} & \textit{bibtex} & \textit{xdvi} & \textit{cd} & \textit{ls} & \textit{less} \\
\textit{emacs} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{latex} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{bibtex} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{xdvi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{cd} & 0 & 0 & 0 & 0 & 1 & 2 & 2 \\
\textit{ls} & 0 & 0 & 0 & 0 & 1 & 1 & 2 \\
\textit{less} & 0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}$$

**Fig. 7** Co-occurrence matrix for $X_2$.

$$\begin{array}{lccccccc}
 & \textit{emacs} & \textit{latex} & \textit{bibtex} & \textit{xdvi} & \textit{cd} & \textit{ls} & \textit{less} \\
\textit{emacs} & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
\textit{latex} & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\textit{bibtex} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{xdvi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{cd} & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
\textit{ls} & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\
\textit{less} & 1 & 1 & 0 & 1 & 0 & 0 & 0
\end{array}$$

**Fig. 8** Co-occurrence matrix for $X_3$.

tance. Matrix $V$ with $K \times N$ dimensions gives the representation of eigenvectors of $A^T A$.

Let us consider three sequences $X_1$, $X_2$, and $X_3$, of six UNIX commands as our running example.

$X_1$ : `emacs emacs latex bibtex latex xdvi`

$X_2$ : `cd ls less cd ls less`.

$X_3$ : `cd ls less emacs latex xdvi`

One can assume that the user edits a file using an emacs application and compiles it with `latex` and `bibtex` commands in the sequence $X_1$. While in the sequence $X_2$, the user browses directories and files using commands such as `cd`, `ls`, and `less` and the user in $X_3$ browses files and directories and edits a file using `emacs` and compiles it with `latex`. The co-occurrence matrices for these three sequences are depicted in **Figs. 6**, **7**, and **8**, respectively. All three co-occurrence matrices are computed with a window size of six and a scope size of three.

We call the eigenvalues computed using PCA the Eigen co-occurrence matrices. The first and the second Eigen co-occurrence matrices of the three sequences are depicted in **Fig. 9** and **Fig. 10**, respectively. Note that the first Eigen co-occurrence matrix shows positive correlations among events `cd`, `ls`, and `less`, and negative correlations among events such as `emacs`, `latex`, `bibtex`, and `xdvi`.

In contrast, the second Eigen co-occurrence

$$\begin{array}{lccccccc}
 & \textit{emacs} & \textit{latex} & \textit{bibtex} & \textit{xdvi} & \textit{cd} & \textit{ls} & \textit{less} \\
\textit{emacs} & -0.23 & -0.27 & -0.23 & 0 & 0 & 0 & 0 \\
\textit{latex} & 0 & -0.23 & -0.23 & -0.27 & 0 & 0 & 0 \\
\textit{bibtex} & 0 & -0.23 & 0 & -0.23 & 0 & 0 & 0 \\
\textit{xdvi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{cd} & 0 & 0 & 0 & 0 & 0.23 & 0.27 & 0.27 \\
\textit{ls} & 0 & 0 & 0 & 0 & 0.23 & 0.23 & 0.27 \\
\textit{less} & 0 & 0 & 0 & 0 & 0.23 & 0.23 & 0.23
\end{array}$$

**Fig. 9** First Eigen cooccurrence matrix.

$$\begin{array}{lccccccc}
 & \textit{emacs} & \textit{latex} & \textit{bibtex} & \textit{xdvi} & \textit{cd} & \textit{ls} & \textit{less} \\
\textit{emacs} & 0.16 & 0 & 0.16 & -0.32 & 0 & 0 & 0 \\
\textit{latex} & 0 & 0.16 & 0.16 & 0 & 0 & 0 & 0 \\
\textit{bibtex} & 0 & 0.16 & 0 & 0.16 & 0 & 0 & 0 \\
\textit{xdvi} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\textit{cd} & -0.32 & 0 & 0 & 0 & 0.16 & 0 & 0 \\
\textit{ls} & -0.32 & -0.32 & 0 & 0 & 0.16 & 0.16 & 0 \\
\textit{less} & -0.32 & -0.32 & 0 & -0.32 & 0.16 & 0.16 & 0.16
\end{array}$$

**Fig. 10** Second Eigen cooccurrence matrix.

matrix shows a mixture of positive and negative correlations among the same set of commands as in the negative correlations of the first Eigen co-occurrence matrix and negative correlations among that of the positive correlations.

### 4.2 Construction of Network Models

The last part of the ECM method is to construct network models, which can be used for detailed analysis of the obtained features. The ECM method offers two ways to construct network models: (1) a combined network (CN) model and (2) a layered network (LN) model. A CN model represents the overall characteristic features observed in the original co-occurrence matrix and a LN model represents its distinct characteristic features.

#### 4.2.1 Combined Network Model

A CN model which we denote $C'$, is a reconstructed co-occurrence matrix from the Eigen co-occurrence matrix space and represents the overall characteristic features of the original co-occurrence matrix $C$. Matrix $C'$ is computed by,

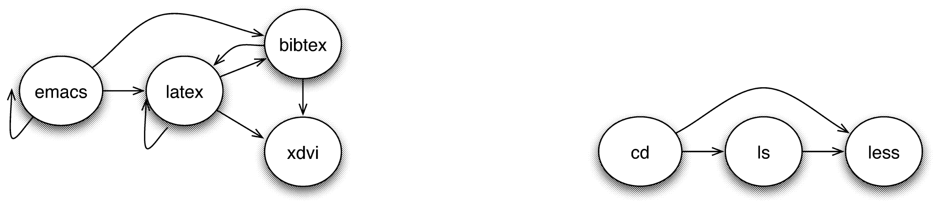
$$C - C_{mean} \simeq \sum_{k=1}^{K} < u_k, a_i > u_k = C', \quad (2)$$

where $< x, y >$ denotes an inner product of $x$ and $y$ and $C'$ can be further separated into $C'_+$ and $C'_-$, the *positive* and *negative* network, respectively. Here the positive (or negative) network provides the characteristic patterns observed for each sequence in terms of the correlation strength in relation to the average co-occurrence matrix of the sequences in the learning dataset.

There may be elements in $C'_+$ (or $C'_-$) that are too small to server as the principal characteristics of $C'$. Thus, instead of using all the $C'_+$

$$\begin{array}{c|ccccccc} & emacs & latex & bibtex & xdvi & cd & ls & less \\ \hline emacs & 1.15 & 1.00 & 1.15 & -0.58 & 0 & 0 & 0 \\ latex & 0 & 1.15 & 1.15 & 1.00 & 0 & 0 & 0 \\ bibtex & 0 & 1.15 & 0 & 1.15 & 0 & 0 & 0 \\ xdvi & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ cd & -0.58 & 0 & 0 & 0 & -0.58 & -1.00 & -1.00 \\ ls & -0.58 & -0.58 & 0 & 0 & -0.58 & -0.58 & -1.00 \\ less & -0.58 & -0.58 & 0 & -0.58 & -0.58 & -0.58 & -0.58 \end{array}.$$

Combined matrix

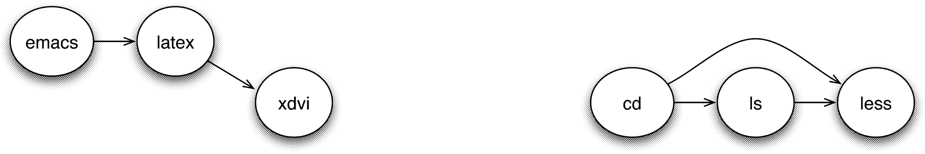


(a) Positive network ($h \geq 1.0$) (b) Negative network ($h \leq -1.0$)

**Fig. 11** Combined network models for sequence $X_1$.

$$\begin{array}{c|ccccccc} & emacs & latex & bibtex & xdvi & cd & ls & less \\ \hline emacs & 0.87 & 1.00 & 0.87 & 0 & 0 & 0 & 0 \\ latex & 0 & 0.87 & 0.87 & 1.00 & 0 & 0 & 0 \\ bibtex & 0 & 0.87 & 0 & 0.87 & 0 & 0 & 0 \\ xdvi & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ cd & 0 & 0 & 0 & 0 & -0.87 & -1.00 & -1.0 \\ ls & 0 & 0 & 0 & 0 & 0.87 & -0.87 & -1.0 \\ less & 0 & 0 & 0 & 0 & 0.87 & -0.87 & -0.87 \end{array}.$$

First layered matrix



(a) Positive network ($h \geq 1.0$) (b) Negative network ($h \leq -1.0$)

**Fig. 12** First layer network models for sequence $X_1$.

(or $C'_{-}$) elements to construct network models, we allow the setting of a threshold $h$ and choose the elements that are larger (or smaller) than $h$ (or $-h$) to construct the network models. Assigning a higher value to $h$ reduces the number of nodes in the network and consequently creates a network model with a different topology.

In **Fig. 11**, the combined network constructed from the example sequence $X_1$ is depicted as an example. The threshold for the network is set to 1.0 for the positive network and $-1.0$ for the negative network. It can be seen that the positive network is composed of commands that appear in the sequence, whereas the negative network is composed of commands that do not. Note that these networks describe the characteristic relations of the sequence $X_1$ in relation to overall features of all the three given sequences, $X_1$, $X_2$, and $X_3$.

### 4.2.2 Layered Network Model

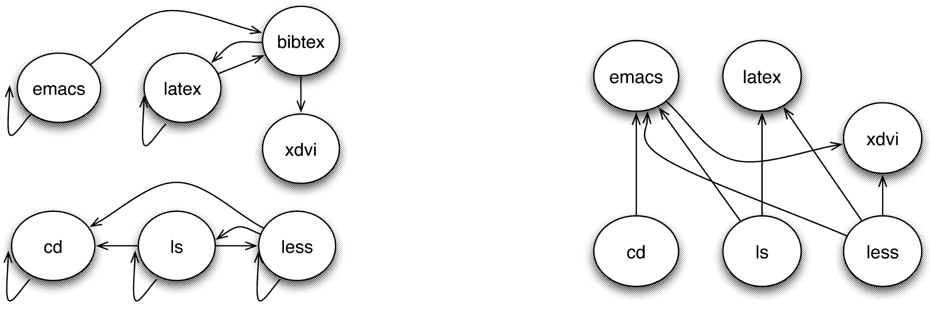
A CN model, while providing the overall characteristic of a sequence, loses the distinct characteristics derived from the Eigen co-occurrence matrices. An LN model can represent such features and is given by

$$C'_k = < u_k, a_i > u_k, \qquad (3)$$

where $C'_k$ denotes an adjacent matrix computed by a co-occurrence matrix projected on the $k$th Eigen co-occurrence matrix of the vector representation of the $i$th co-occurrence matrix $X_i$. As in the LN model, a positive and a negative network can be constructed from $C'_k$. **Figures 12** and **13** are examples of the first and second layers of network models for example sequence $X_1$. It can be observed from the figures that the characteristic features for $X_1$ are mostly derived from the first Eigen co-occurrence matrix and the second Eigen co-occurrence matrix contributes very little to the features of $X_1$.

$$\begin{array}{c|ccccccc}
 & emacs & latex & bibtex & xdvi & cd & ls & less \\
\hline
emacs & 0.29 & 0 & 0.29 & -0.58 & 0 & 0 & 0 \\
latex & 0 & 0.29 & 0.29 & 0 & 0 & 0 & 0 \\
bibtex & 0 & 0.29 & 0 & 0.29 & 0 & 0 & 0 \\
xdvi & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
cd & -0.58 & 0 & 0 & 0 & 0.29 & 0 & 0 \\
ls & -0.58 & -0.58 & 0 & 0 & 0.29 & 0.29 & 0 \\
less & -0.58 & -0.58 & 0 & -0.58 & 0.29 & 0.29 & 0.29
\end{array}.$$

Second layered matrix



(a) Positive network ($h \geq 0.2$) (b) Negative network ($h \leq -0.2$)

**Fig. 13** Second layered network models for sequence $X_1$.

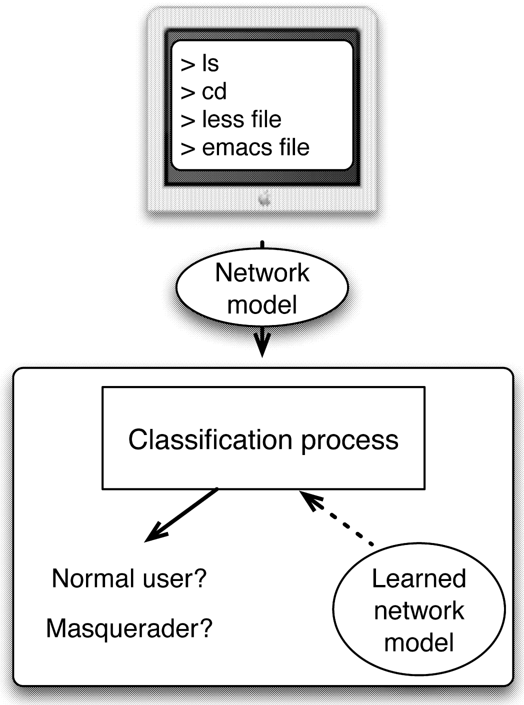
## 5. Anomaly Detection Using ECM Models

Our approach to anomaly detection is based on the idea that anomalous behavior is an unusual activity that will manifest as significant excursions from normal behavior. A common approach to anomaly detection is to first create a *profile* defining a normal user's behavior and then to measure the *similarity* of a current behavior with the crated profile and note any behavior that deviates from the profile. In this section, we describe an application of ECM method to anomaly detection. The anomaly detection procedure follows the general flow depicted in **Fig. 14**.

### 5.1 Procedures

When a sequence $X_t$ is to be tested for whether it is normal or anomalous, we follow this procedure:

(a) Construct a co-occurrence matrix from $X_t$.
(b) Project the obtained co-occurrence matrix on the co-occurrence matrix space, which is computed from a set of normal learning sequences (profile), and obtain its feature vector.
(c) Multiply each element of the feature vector by its corresponding Eigen co-occurrence matrix to obtain network models.
(d) Compute the similarity between network models of $X_t$ and those of the profile.
(e) Classify the testing sequence as anomalous or normal based on a similarity measurement using threshold $\epsilon$.



```
> ls
> cd
> less file
> emacs file
```

**Fig. 14** General flow of system.

### 5.2 Similarity Measurement

The problem of measuring the similarity among network models can be transformed to the problem of matching between networks. However, this problem is known to be NP complete. As a simplified similarity function to reduce the computational cost, we define $r$ connected nodes as a unit of a sub-network and use the number of sub-networks that the two networks have in common as the similarity measurement.

Given a network model $C_x$, we transform it into a vector representation $W_x$, each of whose element $w_{i_1,i_2,\dots,i_r}$ is a binary value (0 or 1) indicating the existence of sub-networks with $r$ connected nodes. The $w_{i_1,i_2,\dots,i_r}$ is defined as,

$$w_{i_1,i_2,\dots,i_r} = w_{i_1+i_2m+\dots+i_rm^{r-1}} = \prod_{k=1}^{r-1} a(i_k, i_{k+1}), \qquad (4)$$

where

$$a(i_k, i_{k+1}) = \begin{cases} 1 & \text{if } a(i_k, i_{k+1}) > h \\ 0 & \text{otherwise,} \end{cases} \qquad (5)$$

where $i_k$ defines the $k$th node of the $r$ connected sub-network whose starting node is the $i$th event $e_i$ in the event set $E$. For example, when $r$ is 2, the vector representation of the matrix in Fig. 3 becomes the vector with 9 dimensions:

$$\begin{array}{ccccccccc} e_1e_1 & e_1e_2 & e_1e_3 & e_2e_1 & e_2e_2 & e_2e_3 & e_3e_1 & e_3e_2 & e_3e_3 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{array}. \qquad (6)$$

Representing sub-network models as a binary vector enables us to compute the similarity by simply taking the product of vectors. Thus, the similarity between two network models $C_a$ and $C_b$ is defined as,

$$sim(C_a, C_b) = < W_a, W_b >, \qquad (7)$$

where $< W_a, W_b >$ indicates the inner product of vectors $W_a$ and $W_b$.

The similarity between given two CN models, $CN_a$ and $CN_b$ can thus be computed by,

$$similarity_{CN} = sim(CN_a, CN_b). \qquad (8)$$

Likewise, the similarity between two LN models, $LN_a$ and $LN_b$, can be computed by,

$$similarity_{LN} = \sum_{k=1}^{N} sim(LN_a^k, LN_b^k), \qquad (9)$$

where $LN^k$ is the $k$th layer network model.

## 6. Experiments

We applied the ECM method to one of the anomaly detection problems, masquerade detection using UNIX command sequences. Masqueraders are intruders masquerading as valid users.

### 6.1 Data

We used a database of 15,000 commands entered at a UNIX prompt for a set of 50 users provided by Schonlau, et al. [11]. The dataset consists of 50 users' commands entered at a UNIX prompt, with the 15,000 recorded commands for each user. Due to privacy concerns, the dataset includes no reporting of the flags, aliases, arguments, or shell grammar. The users are designated as User 1, User 2, and so on. The first 5,000 commands are entered by a legitimate user, and the masquerading commands are inserted in the remaining 10,000 commands. All the user sequences were decomposed into blocks of 100 commands ($w = 100$). **Figure 15** illustrates the composition of the dataset.

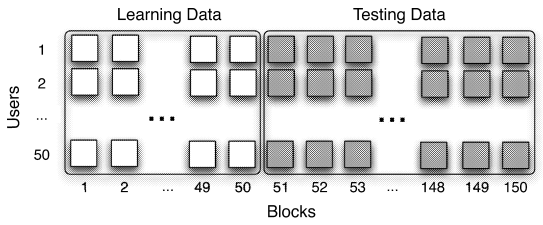


**Fig. 15** Composition of experimental dataset.

### 6.2 Parameter Settings

In the creation of Eigen co-occurrence basis, we took all the users' training data-sets, consisting of 2,500 (50 sequences x 50 users) blocks ($N$ = 2,500). The set of observation events ($E = e_1, e_2, \dots, e_n$) was determined by the unique events appearing in the learning data-set, which accounted for 635 commands ($n$ = 635). Each decomposed block was converted into a co-occurrence matrix with a scope size of six ($s$ = 6). We took 50 Eigen co-occurrence matrices ($K = 50$), whose contribution rate was approximately 90%, and defined this as the co-occurrence matrix space.

The user's profile was created using his training data-set. We first converted all of his training blocks to co-occurrence matrices and obtained the corresponding *positive* and *negative* LN models with a threshold of 0 ($h$ = 0). We only used the *positive* LN models to define each user's profile.

To classify a testing sequence $X_i$ as anomalous or normal, we computed the similarity between LN models of $X_i$ and those of the user profile. If the computed similarity was under a threshold $\epsilon_u$ of the User $u$, then the testing sequence was classified as anomalous; otherwise, it was classified as normal. In the similarity measure for sub-networks, we gave $r$ the value of three ($r = 3$).

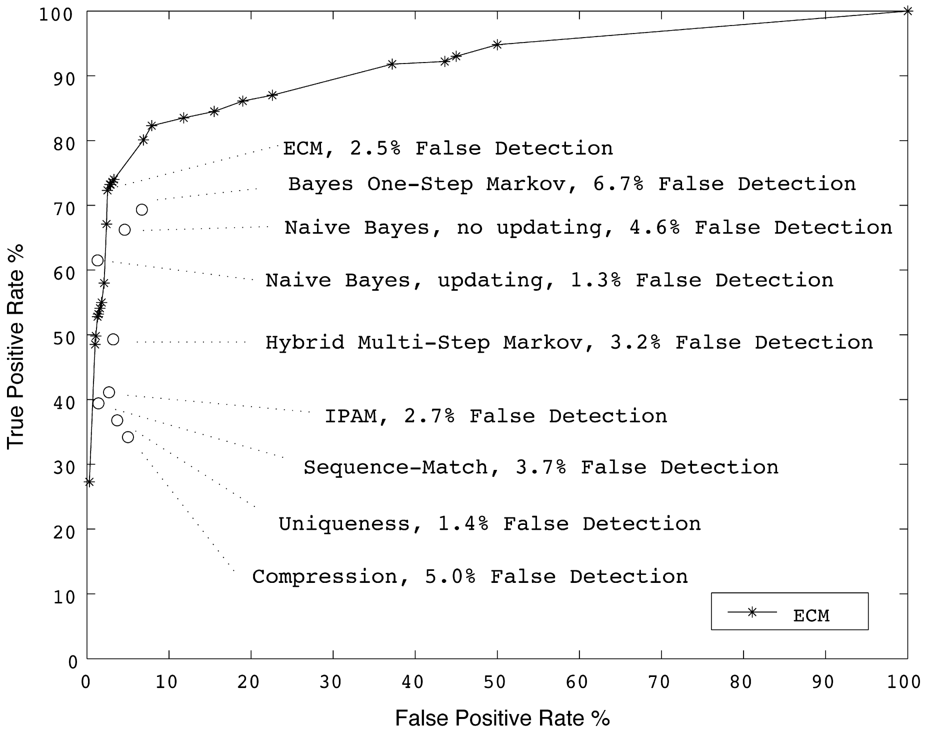


**Fig. 16** ROC curve for the ECM method with the best results from other methods shown for comparison.

### 6.3 Results

The results show the trade-off between the true positives and false positives. A receiver operation characteristic curve (ROC curve) is often used to represent this trade-off. The percentages of true positives and false positives are shown on the y- and the x-axes of an ROC curve, respectively. Any increase in true positive rates will be accompanied by an increase in false positive rates. The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the results are, since this pattern indicates high true positive rates and, correspondingly, low false positive rates.

Schonlau, et al.[11)] and Maxion, et al.[12)] have applied a number of masquerade detection techniques, including Bayes 1-step Markov, hybrid multi-step Markov, IPAM, uniqueness, sequence-match, compression, and naive Bayes, to the same dataset used in this study. (See Refs. 11) and 12) for detailed explanations of each technique.) Their results are shown in **Fig. 16**, along with our results from the ECM method. For the ECM method, we have plotted different true positive rates and false positive rates by changing $\alpha$ in the expression:

$$\epsilon_u^{opt} + \alpha, \qquad (10)$$

where $\epsilon_u^{opt}$ is the optimal threshold for User $u$. The optimal threshold $\epsilon_u^{opt}$ is obtained by finding the largest correct detection rate with a false detection rate of less than $\alpha$%. We set $\alpha$ to 20 in this experiment and used the same values of $\epsilon_u$ throughout all the test sequences (no updating). As a result, the ECM method achieved a 72.3% correct detection rate, with a 2.5% false detection rate. As can be seen from the data, the ECM method achieved one of the best scores among the various approaches.

## 7. Conclusion and Future Work

We presented a method, called ECM, which works as a modeling step before the classification of event sequences. ECM starts with the creation of a co-occurrence matrix, a matrix whose rows and columns correspond to the observation events. The values of the matrix represent the strength of the correlation between two events. ECM then finds the lower-ranked approximation to the co-occurrence matrix, through the use of a Principal Component Analysis (PCA) and consequently, the Eigen co-occurrence matrix space is defined with low-dimensionality space. When a new sequence is issued, it is projected into this low-dimensional space, and a feature vector is created. ECM finally generates so-called combined network and layered networks, which can represent the detailed characteristics of a sequence in terms of frequency (or rarity) in relation to the average sequence of all the sequences that are being examined.

We have shown the ECM method to be effective in detecting masqueraders using the Schonlau dataset. This shows that the principal features from the obtained model of a user behavior are successfully extracted, and that a detailed analysis using layered networks can provide sufficient, useful features for classifications.

Since the ECM method can be used to model any sequence, a lot of interesting work remains. Applying the method to other sequences collected by computer systems, such as system calls issued by programs, network packets, and keystroke logs, would be very interesting.

However, a learning dataset for our method may be very large depending on the type of sequences being considered, i.e. network packets, and thus, building a learning dataset can be computationally costly. Another issue related to building the learning dataset is its maintenance to keep it up-to-date to changes of behavioral pattern of the input data. When installing the system to the actual system, it will be essential to provide schemes for updating the learning dataset. Moreover, the network-matching scheme adopted in the masquerade detection experiments is a pretty rough measure, and our method could benefit from the use of finer measures that could be borrowed from the graph-theory field.

Finally, the concept of constructing network models from Eigen co-occurrence matrices is applicable not only to matrices generated from sequences but also to other types of data, such as medical databases that contain many attributes. In the work we have done [13] we have shown that ECM models can effectively capture characteristic features from medical datasets that that can be used to distinguish healthy patients from diseased ones.

## References

1) Lane, T. and Brodley, C.E.: Temporal sequence learning and data reduction for anomaly detection, *ACM Trans. Information and System Security*, Vol.2, No.3, pp.295–331 (1999).

2) Ye, N., Li, X., Chen, Q., Emran, S.M. and Xu, M.: Probablistic Techniques for Intrusion Detection Based on Computer Audit Data, *IEEE Trans. Systems Man and Cybernetics, Part A* (*Systems & Humans*), Vol.31, No.4, pp.266–274 (2001).

3) Hofmeyr, S.A., Forrest, S. and Somayaji, A.: Intrusion Detection using Sequences of System Calls, *Journal of Computer Security*, Vol.6, No.3, pp.151–180 (1998).

4) Lee, W. and Stolfo, S.J.: A framework for constructing features and models for intrusion detection systems, *ACM Trans. Information and System Security* (*TISSEC*), Vol.3, No.4, pp.227–261 (2000).

5) Sekar, R., Bendre, M. and Bollineni, P.: A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors, *Proc. 2001 IEEE Symposium on Security and Privacy*, Oakland, pp.144–155 (2001).

6) Wagner, D. and Dean, D.: Intrusion Detection via Static Analysis, *Proc. 2001 IEEE Symposium on Security and Privacy*, Oakland, pp. 156–168 (2001).

7) Abe, H., Oyama, Y., Oka, M. and Kato, K.: Optimization of Intrusion Detection System Based on Static Analyses (in Japanese), *IPSJ Trans. Advanced Computing Systems*, pp.11–20 (2004).

8) Kosoresow, A.P. and Hofmeyr, S.A.: A Shape of Self for UNIX Processes, *IEEE Software*, Vol.14, No.5, pp.35–42 (1997).

9) Oka, M., Oyama, Y., Abe, H. and Kato, K.: Anomaly Detection Using Layered Networks Based on Eigen Co-occurrence Matrix, *Proceedings of the Seventh International Symposium on Recent Advances in Intrusion Detection* (*RAID*), pp.223–237 (2004).

10) Warrender, C., Forrest, S. and Pearlmutter, B.A.: Detecting Intrusions Using System Calls: Alternative Data Models, *Proc. IEEE Symposium on Security and Privacy*, pp.133–145 (1999).

11) Schonlau, M., DuMouchel, W., Ju, W.-H., Karr, A.F., Theus, M. and Vardi, Y.: Computer intrusion: Detecting masquerades, *Proc. Statistical Science*, Vol.16, No.1, pp.58–74 (2001).

12) Maxion, R.A. and Townsend, T.N.: Masquerade Detection Using Truncated Command Lines, *Proc. International Conference on Dependable Systems and Networks* (*DSN-02*), pp.219–228 (2002).

13) Oka, M., Koiso, T. and Kato, K.: Extracting Features of Patients using the Eigen Co-occurrence Matrix Algorithm, *Proc. 15th*

*European Conference on Machine Learning* (*ECML*) *and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases* (*PKDD*), pp.86–97 (2004).

(Received August 6, 2006)
(Accepted March 1, 2007)
(Online version of this article can be found in the IPSJ Digital Courier, Vol.3, pp.269–279.)

**Mizuki Oka** received her B.E. and M.E. degrees from the University of Tsukuba, Japan, in 2003 and 2005, respectively. She is currently a graduate student in the Doctoral Program in Graduate School of Systems and Information Engineering at the University of Tsukuba. Her research interests include Image Processing, Kansei Engineering, and Data Mining.

**Kazuhiko Kato** received the B.E. and M.E. degrees from the University of Tsukuba, Japan, in 1985 and 1987, respectively. He received the Ph.D. degree from the University of Tokyo, Japan, in 1992. From 1989 to 1993, he was a research associate in the Department of Information Sciences, Faculty of Sciences at the University of Tokyo. He is currently a professor in the Department of Computer Science, Graduate School of System Information Engineering at the University of Tsukuba. His current research interests include operating systems, secure computing, and autonomous distributed systems.